On Adaptive Security of Delayed-Input Sigma Protocols and Fiat-Shamir NIZKs

On Adaptive Security of Delayed-Input Sigma Protocols and Fiat-Shamir NIZKs*

Michele Ciampi[1][0000−0001−5062−0388], Roberto Parisella[2], and Daniele Venturi[3][0000−0003−2379−8564]

[1] The University of Edinburgh, Edinburgh, UK mciampi@ed.ac.uk
[2] Simula UiB, Bergen, Norway roberto@simula.no
[3] Sapienza University of Rome, Rome, Italy venturi@di.uniroma1.it

Abstract We study *adaptive security* of delayed-input Sigma protocols and non-interactive zero-knowledge (NIZK) proof systems in the common reference string (CRS) model. Our contributions are threefold:

- We exhibit a generic compiler taking any delayed-input Sigma protocol and returning a delayed-input Sigma protocol satisfying *adaptive-input* special honest-verifier zero knowledge (SHVZK). In case the initial Sigma protocol also satisfies *adaptive-input* special soundness, our compiler preserves this property.
- We revisit the recent paradigm by Canetti *et al.* (STOC 2019) for obtaining NIZK proof systems in the CRS model via the Fiat-Shamir transform applied to so-called *trapdoor* Sigma protocols, in the context of adaptive security. In particular, assuming correlation-intractable hash functions for all sparse relations, we prove that Fiat-Shamir NIZKs satisfy either:
  (i) Adaptive soundness (and non-adaptive zero knowledge), so long as the challenge is obtained by hashing both the prover's first round and the instance being proven;
  (ii) Adaptive zero knowledge (and non-adaptive soundness), so long as the challenge is obtained by hashing only the prover's first round, and further assuming that the initial trapdoor Sigma protocol satisfies adaptive-input SHVZK.
- We exhibit a generic compiler taking any Sigma protocol and returning a *trapdoor* Sigma protocol. Unfortunately, this transform does not preserve the delayed-input property of the initial Sigma protocol (if any). To complement this result, we also give yet another compiler taking any delayed-input trapdoor Sigma protocol and returning a delayed-input trapdoor Sigma protocol with adaptive-input SHVZK.

An attractive feature of our first two compilers is that they allow obtaining *efficient* delayed-input Sigma protocols with adaptive security, and *efficient* Fiat-Shamir NIZKs with adaptive soundness (and non-adaptive zero knowledge) in the CRS model. Prior to our work, the latter was only possible using generic NP reductions.

Keywords: Sigma protocols · Non-interactive zero knowledge · Adaptive security.

* Michele Ciampi is the corresponding author and was supported by H2020 project PRIVILEDGE #780477

1 Introduction

Sigma protocols are a special class of three-round public-coin interactive proofs between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$, where $\mathcal{P}$'s goal is to convince $\mathcal{V}$ that a common statement x belongs to a given NP language L. The prover knows a witness w (corresponding to x) as auxiliary input, and starts the interaction by sending a first message a (possibly depending on both x, w); the verifier then sends a uniformly random ℓ-bit challenge c, to which the prover replies with a last message z. Finally, the verifier decides whether $x \in L$ based on x and the transcript (a, c, z). Despite *completeness* (*i.e.*, the honest prover always convinces the honest verifier about true statements), Sigma protocols satisfy two additional properties known as *special soundness* (SS) and *special honest-verifier zero knowledge* (SHVZK). The former is a strong form of *soundness* (*i.e.*, no malicious prover can convince the verifier about the veracity of *false* statements $x \notin L$), which in fact implies that Sigma protocols are proofs of knowledge [35, 6]; the latter requires the existence of an efficient simulator $\mathcal{S}$ that, given any *true* statement $x \in L$ and any possible challenge c, is able to simulate an *honest* transcript (a, c, z) between the prover and the verifier, which in particular implies that honest transcripts do not reveal anything about the witness to the eyes of an honest-but-curious verifier. While Sigma protocols exist for all of NP (as Blum's protocol [11] for Hamiltonian Graphs is a Sigma protocol), the latter comes at the price of expensive NP reductions. Luckily, Sigma protocols also exist for many concrete languages based on number theory and lattices (such as Quadratic Residuosity [35], Discrete Log [50, 47], Factoring [33], and Learning with Errors [51, 45, 2]), and these protocols are very efficient, thus opening the way to a plethora of cryptographic applications, *e.g.* to constructing different kinds of commitment schemes [28, 18, 39, 26, 44, 24] and trapdoor hash functions [7], and for obtaining non-interactive zero-knowledge (NIZK) proofs and digital signatures via the celebrated Fiat-Shamir transform [32, 8, 49]. In this paper we study *adaptive security* for both Sigma protocols and Fiat-Shamir NIZKs.

Delayed-input Sigma Protocols. The classical Sigma protocol by Feige, Lapidot and Shamir [31, 43] for Graph Hamiltonicity (henceforth denoted by FLS) has the special property that the prover can compute the first round of the proof without knowing the graph, so long as it knows the number of vertices ahead of time. In particular, the graph and the corresponding Hamiltonian cycle are only needed to compute the prover's last round. More generally, a Sigma protocol is called *delayed-input* if the prover's first round can be computed given only $n = |x|$ (and without knowing x, w). For such Sigma protocols, the standard definitions of SS and SHVZK may not be sufficient as they do not take into account attackers choosing the statement x adaptively based on a partial transcript (a, c). This limitation may have a negative impact[4] on the applications of delayed-input Sigma protocols, particularly in settings where adaptive

[4] We discuss practical applications where adaptive security is of concern in Section 1.3.

security is required. While, the FLS protocol already satisfies both *adaptive-input* SS and *adaptive-input* SHVZK,[5] the latter is only of theoretical interest. Partially motivated by this shortcoming, Ciampi *et al.* [23] proposed a general transformation for turning any delayed-input Sigma protocol into one satisfying *adaptive-input* SS. This leaves the following open problem. **Q1:** *"Do there exist efficient delayed-input Sigma protocols with adaptive security (i.e., satisfying both adaptive-input SS and adaptive-input SHVZK)?"*

Fiat-Shamir NIZKs. The Fiat-Shamir tranform [32] allows to turn a Sigma protocol into a non-interactive proof system by means of a hash functions h with ℓ-bit output. The idea is for the prover to compute a and z as prescribed by the Sigma protocol, where the challenge c is set to $c := h(a||x)$. One can show that this yields a secure NIZK starting from any Sigma protocol, so long as the hash function h is modelled as a random oracle [8, 30]. Whether security of the Fiat-Shamir transform can be proven without resorting to random oracles has been a question subject of intensive study. Here, the goal is to instantiate the random oracle with a set of efficiently computable hash functions $\mathcal{H} = \{h_k\}$, where the hash key k is made available to all parties in the form of a common reference string (CRS). Unfortunately, several negative results are known in this respect [29, 4, 34, 10], which, however, only exclude the possibility of instantiating the Fiat-Shamir transform starting with *any* Sigma protocol or via black-box reductions to falsifiable assumptions. Indeed, a recent line of research[6] established the above negative results can be circumvented:

- Assuming the initial interactive protocol is a *trapdoor* Sigma protocols [14]. Informally, a trapdoor Sigma protocol is a special Sigma protocol in the CRS model satisfying the following two properties: (i) If the statement x is false, then for every first message a, there is a unique challenge c for which there is an accepting third message z that results in an accepting transcript (a, c, z); (ii) There is a trapdoor associated with the CRS that allows us to efficiently compute this "bad challenge" c from the first message a and the statement x being proven.
- Assuming that $\mathcal{H}$ is a family of correlation-intractable (CI) hash functions [17]. Informally, a family $\mathcal{H}$ satisfies CI w.r.t. some relation R if no efficient attacker given the hash key k can produce an input x such that $(x, h_k(x)) \in R$. CI hash functions w.r.t. broad-enough[7] relations have recently been constructed from a variety of assumptions including program obfuscation [15, 46, 41], strong one-way functions [40], key-dependent message secure encryption [16],

[5] Intuitively, adaptive-input SS guarantees extraction even for transcripts (a, c, z) and (a, c', z') for different (possibly adaptively chosen) statements. Similarly, adaptive-input SHVZK requires the simulator to fake the prover's first message given only $n = |x|$.

[6] This research extends previous results showing that CI is sufficient for proving *soundness* of the Fiat-Shamir transform [29, 5, 38].

[7] In particular, sufficient for proving security of Fiat-Shamir NIZKs without random oracles.

circularly-secure fully-homomorphic encryption [14], LWE [48], and LPN along with DDH/QR/DCR/LWE [13].

A natural question is whether Fiat-Shamir NIZKs obtained via CI hash functions are adaptively secure, *i.e.* whether the non-interactive proof resulting from applying the Fiat-Shamir transform to a trapdoor Sigma protocol satisfies both *adaptive* soundness and *adaptive* zero-knowledge in the CRS model.[8] Canetti *et al.* [14] proved that a slight variant of the FLS protocol directly achieves adaptive security, however, in order to be used in applications, the latter requires expensive NP reductions, and thus results in very inefficient NIZKs. They also provide an efficient instantiation using the classical Sigma protocol for Quadratic Residuosity [35], and more in general starting with any *instance-dependent* trapdoor Sigma protocol (in which the trapdoor is allowed to depend on the statement being proven). Unfortunately, instance-dependent trapdoor Sigma protocols are not sufficient to prove adaptive security of Fiat-Shamir NIZKs, thus leaving the following intriguing open question. **Q2:** *"Do there exist efficient trapdoor Sigma protocols allowing to obtain Fiat-Shamir NIZKs with adaptive security (i.e., satisfying both adaptive soundness and adaptive zero knowledge in the CRS model)?"*

1.1 Our Contributions

In this work, we make progress towards answering the above two open questions in the affirmative. Our first contribution is a general compiler taking any delayed-input Sigma protocol and outputting another delayed-input Sigma protocol (for the same language) with adaptive-input SHVZK. Furthermore, assuming the initial Sigma protocol already satisfies adaptive-input SS, so does the Sigma protocol produced by our compiler. Hence, using the transformation by Ciampi *et al.* [23], we obtain a general compiler which allows to turn any delayed-input Sigma protocol into one with adaptive security, which is a positive answer to **Q1**. Next, we revisit the framework for obtaining adaptively-secure NIZKs via the Fiat-Shamir transform using CI hash functions. In particular, we show the following two results:

- In case the challenge c is obtained by hashing both the prover's first round a and the statement x (*i.e.*, $c = h_k(a||x)$), trapdoor Sigma protocols are sufficient for proving *adaptive soundness* and *non-adaptive zero knowledge* of Fiat-Shamir NIZKs in the CRS model.
- In case the challenge c is obtained by hashing only the prover's first round a (*i.e.*, $c = h_k(a)$), trapdoor Sigma protocols satisfying soundness (which in turn follows by SS) and *adaptive-input* SHVZK are sufficient for proving *non-adaptive soundness* and *adaptive zero knowledge* of Fiat-Shamir NIZKs in the CRS model.

[8] The former means that no malicious prover, given the CRS, can produce a false statement along with an accepting non-interactive proof. The latter means that no malicious verifier, given the CRS, can produce a true statement, along with the corresponding witness, for which a non-interactive proof cannot be simulated in polynomial time given the statement alone.

The fact that hashing both the prover's first message and the statement is essential for obtaining adaptive soundness was already known for the random oracle model [9]. In this vein, our paper confirms this to be sufficient in the plain model as well. Our second contribution is a compiler taking any Sigma protocol and outputting a *trapdoor* Sigma protocol (for the same language). Unfortunately, this compiler does not preserve the delayed-input property of the initial Sigma protocol (if any), and thus, by our result from above, only implies Fiat-Shamir NIZKs with adaptive soundness (but not adaptive zero knowledge). This result can still be interpreted as a partial (positive) answer to **Q2**, as it allows to obtain *efficient* Fiat-Shamir NIZKs with *adaptive soundness* (and non-adaptive zero knowledge) in the CRS model for any language admitting a Sigma protocol. Previously to our work, the latter was possible only using expensive NP reductions. Finally, we also show that any delayed-input trapdoor Sigma protocol can be turned into a delayed-input trapdoor Sigma protocol with adaptive-input SHVZK, which (again by our generalization of [14]) would be sufficient for obtaining Fiat-Shamir NIZKs with adaptive zero knowledge (and non-adaptive soundness) in the CRS model. Unfortunately, the only example we know of a delayed-input trapdoor Sigma protocol is FLS (for which [14] directly proved adaptive security of the corresponding Fiat-Shamir NIZK), and thus we view this more as a conceptual contribution providing a possible path towards obtaining efficient Fiat-Shamir NIZKs with adaptive zero knowledge in the future.

1.2 Technical Overview

Adaptive-input SHVZK. Our first compiler exploits so-called instance-dependent trapdoor commitment (IDTC) schemes. Intuitively, this primitive is parameterized by an NP language L and allows a sender to create a commitment $\mathtt{com}$ (with opening $\mathtt{dec}$) to a message m using a statement x as a label. The main idea is that: (i) In case $x \notin L$ is a *false* statement, the commitment satisfies the standard *binding* property. (ii) In case $x \in L$ is a *true* statement, the commitment satisfies the standard *hiding* property and additionally, given a valid witness w for x, one can generate a fake commitment $\mathtt{com}$ that is distributed like an honest commitment but that can later be opened to any message (the so-called *trapdoorness* property). It is well known that IDTCs for any language L can be constructed in a black-box way given any Sigma protocol for L [27, 26, 44, 24].

We now explain how to compile any delayed-input Sigma protocol Σ for a language L into a delayed-input Sigma protocol Σ'' for L that satisfies adaptive-input SHVZK. The transformation relies on an IDTC Π for the language L_{DH} of Diffie-Hellman (DH) tuples, and on a Sigma protocol Σ' for the complement language $\overline{L}_{DH}$ of non-DH tuples.[9]

- The prover, given only $x \in L$, starts by sampling a random non-DH tuple $T \in \overline{L}_{DH}$, along with the corresponding witness, and then computes a commitment $\mathtt{com}$ (with decommitment $\mathtt{dec}$) to the first round a of Σ using T as

[9] We refer the reader to the full version for a description of Σ'.

label. Next, it computes the first round a' of Σ' and forwards $(\mathtt{com}, a', T)$ to the verifier.
- The verifier sends a random ℓ-bit challenge c to the prover.
- Upon receiving a valid witness w for x, the prover computes the third round z and z' of both Σ and Σ', and forwards them to the verifier along with the opening $(a, \mathtt{dec})$ of commitment $\mathtt{com}$.

The proof of (adaptive-input) SS of Σ'' follows readily from the (adaptive-input) SS of Σ and the binding property of Π. Hence, we here focus on the proof of adaptive-input SHVZK. The simulator proceeds as follows:
- Upon receiving challenge $c \in \{0,1\}^\ell$, the simulator first samples a random DH tuple $T \in L_{DH}$ and generates a fake commitment $\mathtt{com}$ using T and its corresponding witness. Next, it runs the SHVZK simulator of Σ' upon input T and c obtaining (a', z') and returns a simulated first round $a'' = (\mathtt{com}, a', T)$.
- Upon receiving statement $x \in L$, the simulator runs the SHVZK simulator of Σ upon input x and c obtaining (a, z). Hence, it opens the commitment $\mathtt{com}$ to a obtaining decommitment $\mathtt{dec}$, and returns a simulated third round $z'' = (z, z', (a, \mathtt{dec}))$.

In the proof, we first move to a mental experiment with a modified simulator that generates (a, z) using the real prover of Σ; this is possible thanks to the SHVZK property of Σ. Next, we replace $T \in L_{DH}$ with $T \in \overline{L}_{DH}$ and $(\mathtt{com}, \mathtt{dec})$ with an honestly computed commitment to a. The DDH assumption and the trapdoorness property of Π imply that no efficient distinguisher can notice such a change. Finally, we use the SHVZK property of Σ' to compute (a', z') as the real prover of Σ' would do, which yields exactly the same distribution of proofs as generated by our compiler, and thus concludes the proof. Note that, besides running Σ, our transformation requires to run an IDTC in parallel with Σ' (to prove that a tuple is DH). The cost of running the IDTC corresponds to running a Sigma protocol for DH tuples. The cost of running a Sigma protocol that proves that a tuple is DH is of 2 exponentiations for the prover and 4 exponentiations for the verifier. Therefore, our compiler adds an overhead of 4 exponentiations for the prover and 8 exponentiations for the verifier.

Adaptive Security of Fiat-Shamir NIZKs. We start by recalling the notion of trapdoor Sigma protocols in more details. Intuitively, a trapdoor Sigma protocol is a special kind of three-round public-coin proof system in the CRS model[10] with the guarantee that, for every valid CRS ω, every false statement $x \notin L$, and every first round a, there is *at most one* challenge $c := f(\omega, a, x)$ such that, for some z, the transcript (a, c, z) is accepting w.r.t. (ω, x). The function f is called the *bad-challenge function*. Moreover, it is possible to generate an honest-looking CRS ω along with a trapdoor τ which allows to efficiently compute the bad challenge c given the first round a and the statement x.

[10] The latter means that, at setup, a CRS ω is generated and distributed to both the prover and the verifier.

Let Σ be a trapdoor Sigma protocol for language L, and Π be the non-interactive proof derived from Σ via the Fiat-Shamir transform using a CI hash family $\mathcal{H}$ for all "efficiently searchable" sparse relations. The proof of adaptive security of Π follows closely the approached used in [14], with a few crucial differences. In particular:

- To show adaptive soundness, one first argues that any prover which, given an honestly-computed CRS (ω, k), is able to produce a statement $x \notin L$ and a non-interactive proof $\pi = (a, z)$ such that (a, c, z) is accepting w.r.t. (ω, x) for $c = h_k(a||x)$ with non-negligible probability, must do so even in case the CRS ω of Σ is generated along with the trapdoor τ. The latter, however, contradicts the CI property of the hash family $\mathcal{H}$ w.r.t. the (efficiently searchable) relation $R_{\omega,\tau} := \{(a||x, c) : \ x \notin L \wedge c = f(\tau, \omega, a, x)\}$, which is easily seen to be sparse thanks to the soundness property of Σ. The key observation that allows to prove adaptive security here is the fact that the hash function takes also x as input, which allows the reduction to CI to go through without knowing x in advance.
- The simulator for adaptive zero knowledge picks a random challenge c and then obtains a invoking the adaptive-input SHVZK simulator of Σ. Hence, it samples a fresh CRS ω and a random hash key k from the conditional distribution $h_k(a) = c$, yielding a simulated CRS (ω, k). Finally, upon receiving $x \in L$ from the adversary, it obtains z from the adaptive-input SHVZK simulator and outputs a simulated proof $\pi = (a, z)$. We note that for this to work it is essential that the challenge c is obtained by hashing only the prover's first message a, as otherwise the simulator would not be able to sample k uniformly from the conditional distribution $c = h_k(a||x)$ without being given x in advance.

From Sigma protocols to trapdoor Sigma protocols. Let us now explain our compiler for turning any Sigma protocol Σ into a *trapdoor* Sigma protocol Σ'. The CRS ω' of Σ' consists of the CRS ω of Σ (if any), along with the public key pk of a (committing) public-key encryption (PKE) scheme. For simplicity, let us assume that the challenge space of Σ is $\{0, 1\}$; it is immediate to extend the challenge space arbitrarily using parallel repetition. The prover of Σ' simply obtains a by running the prover of Σ. Hence, it computes both answers z_0 and z_1 corresponding to the two possible challenges $c = 0$ and $c = 1$, and it encrypts z_0 and z_1 under pk obtaining two ciphertexts e_0, e_1. The prover's first message consists of $a' = (a, e_0, e_1)$. Finally, upon receiving a challenge c, the prover's last message z' consists of the response z_c along with the random coins r_c used to obtain e_c. The verifier accepts if and only if (a, c, z_c) is a valid transcript w.r.t. (ω, x), and additionally (z_c, r_c) is consistent with e_c. It is not hard to show that the above transformation preserves both SS and SHVZK of the underlying protocol Σ, so long as the PKE scheme is semantically secure. To prove that Σ' is a trapdoor Sigma protocol it remains to show how to efficiently compute the bad-challenge function. The main idea here is to let the secret key sk corresponding to pk be the trapdoor τ. This way, given a' and x, we can decrypt e_0, e_1

obtaining[11] the responses z_0, z_1. Note that in case both transcripts $(a, 0, z_0)$ and $(a, 1, z_1)$ are accepting w.r.t. (ω, x), the SS property of Σ implies that $x \in L$. On the other hand, if $x \notin L$, there exists at most one challenge c such that (a, c, z_c) is accepting, and we can determine c efficiently by simply running the verifier algorithm upon input both transcripts $(a, 0, z_0)$ and $(a, _1, z_1)$. Note that, besides running Σ, our transformation requires to encrypt two values using a public-key encryption scheme. Moreover, if we want a trapdoor Sigma protocol with challenge space $\{0,1\}^\ell$ for some $\ell \in \mathbb{N}$, and consequently better soundness, we need to repeat our protocol in parallel ℓ times. If the cost of running Σ is C_P for the prover and C_V for the verifier, and the cost of computing an encryption is C_E, then the cost of running our protocol for the prover is $(C_P + 2C_E)\ell$ and for the verifier is $(C_V + C_E)\ell$.

Adding adaptive-input SHVZK. Note that Σ' as defined above is inherently not delayed-input, even assuming Σ is delayed-input. This is because the prover needs the witness in order to compute the two possible responses z_0, z_1 already in the first round. Our last compiler overcomes this problem by extending our very first transform (for obtaining delayed-input Sigma protocols with adaptive-input SHVZK) to *trapdoor* Sigma protocols. The main idea is to work in the CRS model and replace the IDTC with an *extractable* IDTC (a new notion that we introduce). Intuitively, the difference between IDTCs and extractable IDTCs is that the latter are defined in the CRS model, and the CRS can be generated together with a trapdoor in such a way that, given a commitment of the extractable IDTC scheme with respect to a false instance, it is possible to extract the committed value (which is unique) using the trapdoor. Moreover, the commitment procedure now outputs a message `com` and an instance T such that the verifier can check if the first two components of T are consistent with the CRS. As we show, the latter allows to preserve the trapdoorness property of Σ when applying the transformation described before, while at the same time boosting SHVZK to adaptive-input SHVZK. Finally, we give a simple construction of an extractable IDTC Π for the language L_{DH} of DH tuples. This construction is based on the observation that the classical Sigma protocol for DH tuples has a special extractor which, on input the first round $a = (g^r, h^{r'})$ and γ such that $h = g^\gamma$, outputs the only possible challenge c that would make the transcript (a, c, z) accepting w.r.t. a non-DH tuple (for some z). Given a Sigma protocol Σ for L_{DH}, we then show how to obtain an extractable IDTC. The main idea is to set the CRS to $(g, h = g^\gamma)$ and the trapdoor to $\tau = \gamma$. Each commitment `com` is then equipped with a value $T = (g^\alpha, h^\beta)$ with $\alpha \neq \beta$. Note that in this case (ω, T) corresponds to a non-DH tuple, hence the extractor can be run on `com`, which corresponds to the first round of Σ.

1.3 Applications

Our results directly allow to achieve adaptive security of delayed-input Sigma protocols and Fiat-Shamir NIZKs. Since applications of the latter are well known,

[11] Due to the committing property of the PKE scheme.

below we elaborate on the impact of our results on applications of the former. The delayed-input property directly improves the round complexity of any cryptographic protocol consisting of the following two steps: (1) an NP-statement x and a witness w is defined via an interactive process; and (2) one of the parties involved in the protocol provides a proof that x is a true statement. Indeed, using a delayed-input Sigma protocol that allows proving the validity of x, it is possible to parallelize the above two steps, thus decreasing the round complexity of the overall process. Furthermore, the delayed-input property of FLS has proven to be particularly powerful for providing round-efficient constructions from general assumptions, such as: 4-round (optimal) secure 2PC where only one player gets the output (5 rounds when both players get the output) [42], 4-round 2PC in the simultaneous message exchange model where both parties get the output [22], 4-round MPC for any functionality [19, 3, 1, 12], 3-round non-malleable commitments [20, 36] and 4-round non-malleable commitments [21, 37]. In many cryptographic applications, one party needs to prove an OR statement of the form *"either x is true or I know a trapdoor"*, where neither x nor the trapdoor might be known at the beginning of the protocol. Our adaptive-input SHVZK Sigma protocols can be used to prove exactly this kind of statements, as we can combine adaptive-input (and non-adaptive-input) SHVZK Sigma protocols using the well-known OR composition technique of [25], which yields an adaptive witness-indistinguishable (WI) Sigma protocol (*i.e.*, a Sigma protocol that retains the WI property even when the statement is adaptively chosen after the first round). The notion of adaptive WI was formalized in [23], where the authors proposed a general compiler to obtain this property. The advantage of our approach is that we obtain a more efficient compiler. Indeed, the compiler of [23] requires to compute at least one additional commitment for each statement that composes the OR theorem.

2 Preliminaries

We assume familiarity with the notions of negligible functions, computational indistinguishability, and public-key encryption. We refer to the full version for additional standard definitions. We start the section by introducing our notation. For a string x, we denote its length by $|x|$; if S is a set, $|S|$ represents the number of elements in S. When x is chosen randomly in S, we write $x \leftarrow\!\!\$\ S$. When $\mathcal{A}$ is a randomized algorithm, we write $y \leftarrow\!\!\$\ \mathcal{A}(x)$ to denote a run of $\mathcal{A}$ on input x (and implicit random coins r) and output y; the value y is a random variable, and $\mathcal{A}(x;r)$ denotes a run of $\mathcal{A}$ on input x and randomness r. An algorithm $\mathcal{A}$ is *probabilistic polynomial-time* (PPT) if $\mathcal{A}$ is randomized and for any input $x, r \in \{0,1\}^*$ the computation of $\mathcal{A}(x;r)$ terminates in a polynomial number of steps (in the size of the input). A *polynomial-time* relation R is a relation for which membership of (x,w) w.r.t. R can be decided in time polynomial in $|x|$. If $(x,w) \in R$ then we say that w is a *witness* for *instance* x. A polynomial-time relation R is naturally associated with the NP language L_R defined as $L_R = \{x : \exists w \text{ s.t. } (x,w) \in R\}$. (When R is clear from the context, we simply

write L.) Similarly, an NP language is naturally associated with a polynomial-time relation. We denote by $\hat{L}_R$ the language such that $L_R \subseteq \hat{L}_R$ and membership in $\hat{L}_R$ may be tested in polynomial time.

Sigma Protocols. Let L be an NP language, with corresponding relation R. A Sigma protocol $\Sigma = (\mathcal{P}, \mathcal{V})$ for R is a 3-round public-coin protocol. In particular, an execution of Σ proceeds as follows:

- The prover $\mathcal{P}$ computes the first message using as input the instance to be proved $x \in L$ with the corresponding witness w, and outputs the first message a with an auxiliary information st; we denote this action with $(a, st) \leftarrow\!\!\$\ \mathcal{P}(x, w)$.
- The verifier $\mathcal{V}$, upon receiving a, sends a random string $c \leftarrow\!\!\$\ \{0,1\}^\ell$ with $\ell \in \mathbb{N}$.
- The prover $\mathcal{P}$, upon input c and st, computes and sends z to $\mathcal{V}$; we denote this action with $z \leftarrow\!\!\$\ \mathcal{P}(st, c)$.
- The verifier $\mathcal{V}$, upon input (x, a, c, z), outputs 1 to accept and 0 to reject; we denote this action with $\mathcal{V}(x, a, c, z) = d$ where $d \in \{0, 1\}$ denotes whether $\mathcal{V}$ accepts or not.

Definition 1 (Sigma protocol [25]). *A 3-move protocol Σ with challenge length $\ell \in \mathbb{N}$ is a* Sigma protocol *for a relation R if it enjoys the following properties:*

- ***Completeness**. If $(x, w) \in R$, then all honest 3-move transcripts for (x, w) are accepting.*
- ***Special soundness**. There exists an efficient algorithm $\mathcal{K}$ that, on input two accepting transcripts (a, c, z) and (a, c', z') for x with $c' \neq c$ (we refer to such two accepting transcripts as a* collision*) outputs a witness w such that $(x, w) \in R$.*
- ***Special honest-verifier zero knowledge (SHVZK)**. There exists a PPT simulator algorithm $\mathcal{S}$ that takes as input security parameter 1^λ, $x \in L$ and $c \in \{0,1\}^\ell$, and outputs an accepting transcript for x where c is the challenge (we denote this action with $(a, z) \leftarrow\!\!\$\ \mathcal{S}(x, c)$). Moreover, for all ℓ-bit strings c, the distribution of the output of the simulator on input (x, c) is computationally indistinguishable from the distribution of the 3-move honest transcript obtained when $\mathcal{V}$ sends c as challenge and $\mathcal{P}$ runs on common input x and any private input w such that $(x, w) \in R$.*

The DDH Assumption. We give a high level overview on the DDH assumption and its variants. We refer the reader to the full version for a more formal and complete treatment. Let $\mathcal{G}$ be a cyclic group with generator g, and let A, B and X be elements of $\mathcal{G}$. We say that (g, A, B, X) is a *Diffie-Hellman tuple* (a *DH tuple*, in short) if $A = g^\alpha, B = g^\beta$ for some integers $0 \leq \alpha, \beta \leq |\mathcal{G}| - 1$, and $X = g^{\alpha\beta}$. If this is not the case, (g, A, B, X) is called a *non-DH tuple*.

The *Decisional Diffie-Hellman* (DDH) assumption posits the hardness of distinguishing a randomly selected DH tuple from a randomly selected non-DH

tuple. Consider now the polynomial-time relation $R_{1nDH} := \{((g, A, B, X), \alpha) : A = g^\alpha \text{ and } X = g \cdot B^\alpha\}$. A *1-non-DH* tuple is a tuple $T = (g, A, B, X)$ such that $A = g^\alpha$, $B = g^\beta$ and $X = g \cdot B^\alpha = g^{\alpha \cdot \beta + 1}$. Under the DDH assumption random 1-non-DH tuples are indistinguishable from random non-DH tuple. As showed in [23], a Sigma protocol Σ_{1nDH} for the relation R_{1nDH} can be constructed based on the Sigma protocol Σ_{DH} of [25] to prove that a given tuple is DH. The compiler of [23] is almost as efficient as Σ_{DH} and works as follows. On input tuples (g, A, B, X), the prover and the verifier construct tuples (g, A, B, Y) by setting $Y = X/g$. Then, they simply run Sigma protocol Σ_{DH} upon input the theorem (g, A, B, Y).

2.1 Instance-Dependent Trapdoor Commitment

An *instance-dependent trapdoor commitment* scheme for polynomial-time relation R with message space M is a quadruple of PPT algorithms $(\mathsf{Com}, \mathsf{Dec}, \mathsf{Fake}_1, \mathsf{Fake}_2)$ specified as follows:

- Com is the randomized *commitment* algorithm that takes as input an instance $x \in \hat{L}$ and a message $m \in M$, and outputs *commitment* $\mathtt{com}$ and *decommitment* $\mathtt{dec}$;
- Dec is the *verification* algorithm that takes as input $x \in \hat{L}$, $\mathtt{com}, \mathtt{dec}$ and $m \in M$, and decides whether m is the decommitment of $\mathtt{com}$;
- Fake_1 takes as input $(x, w) \in R$ and outputs *commitment* $\mathtt{com}$, and *equivocation information* $\mathtt{rand}$;
- Fake_2 takes as input $(x, w) \in R$, message $m \in M$, and $(\mathtt{com}, \mathtt{rand})$, and outputs $\mathtt{dec}$;

Definition 2 (Instance-dependent trapdoor commitment scheme). *Let R be a polynomial-time relation. We call $\Pi = (\mathsf{Com}, \mathsf{Dec}, \mathsf{Fake}_1, \mathsf{Fake}_2)$ an* instance-dependent trapdoor commitment *scheme (an* IDTC*, in short) for R if it enjoys the following properties:*

- ***Correctness.*** *For all $x \in \hat{L}$, and all $m \in M$, it holds that*

$$\mathbb{P}\left[\mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1 : \ (\mathtt{com}, \mathtt{dec}) \leftarrow\!\!\$\ \mathsf{Com}(x, m)\right] = 1.$$

- ***Binding.*** *For all $x \notin L$, and for every commitment $\mathtt{com}$, there exists at most one message $m \in M$ for which there is a valid decommitment $\mathtt{dec}$ (*i.e. $\mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1$*).*
- ***Hiding.*** *For every $x \in L$, and every $m_0, m_1 \in M$, the two ensembles $\{\mathtt{com} : (\mathtt{com}, \mathtt{dec}) \leftarrow\!\!\$\ \mathsf{Com}(1^\lambda, x, m_0)\}_{\lambda \in \mathbb{N}}$ and $\{\mathtt{com} : (\mathtt{com}, \mathtt{dec}) \leftarrow\!\!\$\ \mathsf{Com}(1^\lambda, x, m_1)\}_{\lambda \in \mathbb{N}}$ are identically distributed.*
- ***Trapdoorness.*** *For all $(x, w) \in R$ and $m \in M$ the following two distributions coincide:*

$$\{(\mathtt{com}, \mathtt{dec}) : \ (\mathtt{com}, \mathtt{rand}) \leftarrow\!\!\$\ \mathsf{Fake}_1(x, w); \mathtt{dec} \leftarrow\!\!\$\ \mathsf{Fake}_2(x, w, m, \mathtt{com}, \mathtt{rand})\}$$
$$\{(\mathtt{com}, \mathtt{dec}) : \ (\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(x, m)\}.$$

An IDTC can be easily constructed from any Sigma protocol as shown in [26, 39, 28, 23].

2.2 Correlation-Intractable Hash Families

Definition 3 (Hash family). *For a pair of efficiently computable functions $(n(\cdot), m(\cdot))$, a hash family with input length n and output length m is a collection $\mathcal{H} = \{h_k : \{0,1\}^{n(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{\lambda \in \mathbb{N}, k \in \{0,1\}^{s(\lambda)}}$ of keyed hash functions, along with a pair of PPT algorithms specified as follows: (i) $\mathcal{H}.\mathsf{Gen}(1^\lambda)$ outputs a hash key $k \in \{0,1\}^{s(\lambda)}$; (ii) $\mathcal{H}.\mathsf{Hash}(k, x)$ computes the function $h_k(x)$.*

Definition 4 (Correlation intractability). *For a given relation ensemble $R := \{R_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}\}$, a hash family $\mathcal{H} = \{h_k : \{0,1\}^{n(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{\lambda \in \mathbb{N}, k \in \{0,1\}^{s(\lambda)}}$ is said to be R-correlation intractable with security (σ, δ) if for every σ-size attacker $\mathcal{A} := \{\mathcal{A}_\lambda\}$:*

$\mathbb{P}\left[(x, h_k(x)) \in R_\lambda : \; k \leftarrow\!\!\$ \, \mathcal{H}.\mathsf{Gen}(1^\lambda); x \leftarrow\!\!\$ \, \mathcal{A}(k)\right] = O(\delta(\lambda))$.

We say that $\mathcal{H}$ is R-correlation intractable if it is R-correlation intractable with security $(\lambda^c, \lambda^{-c})$ for all constants $c > 1$.

Correlation intractability is a useful and versatile property of random oracles that we would like to guarantee in the standard model. However, even a random oracle is only R-correlation intractable for so-called *sparse* relations.

Definition 5 (Sparsity). *For any relation ensemble $R := \{R_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}\}_\lambda$, we say that R is $\rho(\cdot)$-sparse if for all $\lambda \in \mathbb{N}$ and for any $x \in \{0,1\}^{n(\lambda)}$ it holds that $(x, y) \in R_\lambda$ with probability at most $\rho(\lambda)$ over the choice of $y \leftarrow\!\!\$ \, \{0,1\}^{m(\lambda)}$. When ρ is a negligible function, we say that R is sparse.*

Efficiently Searchable Relations. In this work, we will need hash families achieving correlation intractability for relations R with a unique output $y = f(x)$ associated to each input x, and such that $y = f(x)$ is an efficiently computable function of x.

Definition 6 (Unique output relation). *We say that a relation R is a unique output relation if for every input x, there exists at most one output y such that $(x, y) \in R$.*

Definition 7 (Efficiently searchable relation). *We say that a (necessarily unique-output) relation ensemble R is searchable in (non-uniform) time t if there exists a function $f = f_R : \{0,1\}^* \to \{0,1\}^*$ computable in (non-uniform) time t such that for any input x, if $(x, y) \in R$ then $y = f(x)$; that is, $f(x)$ is the unique y such that $(x, y) \in R$, provided that such a y exists. We say that R is efficiently searchable if it is searchable in time poly(n).*

Programmability. The following property turns out to be very useful in order to prove the zero-knowledge property of non-interactive proofs derived using correlation-intractable hash families.

Definition 8 (1-universality). *We say that a hash family $\mathcal{H}$ is 1-universal if for any $\lambda \in \mathbb{N}$, input $x \in \{0,1\}^{n(\lambda)}$, and output $y \in \{0,1\}^{m(\lambda)}$, we have $\mathbb{P}\left[h_k(x) = y : k \leftarrow\!\!\$ \, \mathcal{H}.\mathsf{Gen}(1^\lambda)\right] = 2^{-m(\lambda)}$.*

We say that a hash family $\mathcal{H}$ is programmable if it is 1-universal, and if there exists an efficient sampling algorithm $\mathsf{Sample}(1^\lambda, x, y)$ that samples from the conditional distribution $k \leftarrow\!\!\$ \, \mathcal{H}.\mathsf{Gen}(1^\lambda) | h_k(x) = y$.

2.3 Non-Interactive Argument Systems

Definition 9 (NIZK argument systems). *A* non-interactive zero-knowledge argument system (NIZK) *for an NP-language L consists of three PPT machines $\Pi := (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$, that have the following properties:*

- **Completeness.** *For all $\lambda \in \mathbb{N}$, and all $(x, w) \in R$, it holds that:* $\mathbb{P}\left[\mathcal{V}(\omega, x, \mathcal{P}(\omega, x, w)) = 1 : \omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, 1^{|x|})\right] = 1$.
- **Soundness.** *For all PPT provers $\mathcal{P}^*$, there exists a negligible function $\nu : \mathbb{N} \to [0,1]$, such that for all $\lambda \in \mathbb{N}$ and for all $x \notin L$:* $\mathbb{P}\left[\mathcal{V}(\omega, x, \pi) = 1 : \omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, 1^{|x|}); \pi \leftarrow\!\!\$\ \mathcal{P}^*(\omega)\right] \leq \nu(\lambda)$.
- **Zero knowledge.** *There exists a PPT simulator $\mathcal{S}$ such that for every $(x, w) \in R$, the distribution ensembles $\{(\omega, \pi) : \omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, 1^{|x|}); \pi \leftarrow\!\!\$\ \mathcal{P}(\omega, x, w)\}_{\lambda\in\mathbb{N}}$ and $\{\mathcal{S}(1^\lambda, x)\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable.*

A NIZK argument system can also satisfy various stronger properties. We list them below.

- **Adaptive zero knowledge.** *For all PPT verifiers $\mathcal{V}^*$ there exists a PPT simulator $\mathcal{S} := (\mathcal{S}_0, \mathcal{S}_1)$ such that the following distribution ensembles are computationally indistinguishable:*
  $\{(\omega, \pi) : \omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, 1^{|x|}); (x, w) \leftarrow\!\!\$\ \mathcal{V}^*(\omega); \pi \leftarrow\!\!\$\ \mathcal{P}(\omega, x, w); (x, w) \in R\}_{\lambda\in\mathbb{N}}$
  $\{(\omega, \pi) : (\omega, \tau) \leftarrow\!\!\$\ \mathcal{S}_0(1^\lambda, 1^{|x|}); (x, w) \leftarrow\!\!\$\ \mathcal{V}^*(\omega); \pi \leftarrow\!\!\$\ \mathcal{S}_1(\omega, \tau, x); (x, w) \in R\}_{\lambda\in\mathbb{N}}$
- **Adaptive soundness.** *For all PPT prover $\mathcal{P}^*$, there exists a negligible function $\nu : \mathbb{N} \to [0,1]$, such that for all $\lambda \in \mathbb{N}$:* $\mathbb{P}\left[\mathcal{V}(\omega, x, \pi) = 1 : \omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, 1^{|x|}); (x, \pi) \leftarrow\!\!\$\ \mathcal{P}^*(\omega); x \notin L\right] \leq \nu(\lambda)$.

3 A Compiler for Adaptive-input HVZK

Definition 10 (Delayed-input protocols [23]). *A* delayed-input *three-move protocol for polynomial-time relation R is a three-move protocol $(\mathcal{P}, \mathcal{V})$ in which the first message of $\mathcal{P}$ can be computed on input the length n of the common theorem in unary notation.*[12]

Definition 11 (Adaptive-input special soundness). *A delayed-input 3-round protocol $\Sigma = (\mathcal{P}, \mathcal{V})$ for relation R enjoys* adaptive-input special soundness *if there exists a polynomial-time algorithm $\mathcal{K}$ such that, for any $x_1, x_2 \in L$, and for any pair of accepting transcripts (a, c_1, z_1) for input x_1 and (a, c_2, z_2) for input x_2 with $c_1 \neq c_2$, outputs witnesses w_1 and w_2 such that $(x_1, w_1) \in R$ and $(x_2, w_2) \in R$.*

Definition 12 (Adaptive-input SHVZK). *A delayed-input 3-round protocol $\Sigma = (\mathcal{P}, \mathcal{V})$ for relation R satisfies* adaptive-input special honest-verifier zero-knowledge (adaptive-input SHVZK) *if there exists a PPT simulator algorithm $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ such that for all PPT adversaries $\mathcal{A}$ and for all challenges $c \in \{0,1\}^\ell$ there is a negligible function $\nu : \mathbb{N} \to [0,1]$ for which $\left|\mathbb{P}[b' = b] - \frac{1}{2}\right| \leq \nu(\lambda)$ in the following game:*

[12] For simplicity, in what follows, we sometimes drop input 1^n when describing the prover of a delayed-input Sigma protocol.

1. *The challenger sends* (a, c) *to* $\mathcal{A}$*, where the value* a *is either computed using* $(a, st) \leftarrow\!\!\$\ \mathcal{P}(1^\lambda, 1^n)$ *(in case* $b = 0$*) or* $(a, st) \leftarrow\!\!\$\ \mathcal{S}_0(1^\lambda, 1^n, c)$ *(in case* $b = 1$*).*
2. *The adversary* $\mathcal{A}$ *sends a pair* (x, w) *to the challenger, where* $|x| = n$*. Hence, if* $(x, w) \in R$*, the challenger sends* z *to* $\mathcal{A}$*, where the value* z *is either computed using* $z \leftarrow\!\!\$\ \mathcal{P}(x, w, st, c)$ *(in case* $b = 0$*) or* $z \leftarrow\!\!\$\ \mathcal{S}_1(x, st)$ *(in case* $b = 1$*); Else, the challenger sends* $z = \perp$ *to* $\mathcal{A}$*.*
3. *The adversary* $\mathcal{A}$ *outputs a bit* b'*.*

The Transformation. It turns out that the celebrated Sigma protocol by Lapidot and Shamir [43] is already delayed-input, and moreover it satisfies both adaptive-input special soundness[13] and adaptive-input SHVZK. While this protocol works for any NP relation, it is very inefficient as it requires generic NP reductions. Hence, it is a natural question whether there are efficient Sigma protocols that are delayed-input and satisfy both adaptive-input special soundness and SHVZK. A partial answer to this question was given in [23], which shows how to transform a large class of delayed-input Sigma protocols into ones with adaptive-input special soundness. In this section, we give yet another transform that allows to turn *any* delayed-input Sigma protocol into one satisfying adaptive-input SHVZK. Moreover, assuming the initial Sigma protocol already satisfies adaptive-input special soundness, our transformation preserves this property. Let Σ be a delayed-input Sigma protocol for a polynomial-time relation R. We construct a Sigma protocol Σ'' for R based on the following additional building blocks: (i) An IDTC $\Pi = (\mathsf{Com}, \mathsf{Dec}, \mathsf{Fake}_1, \mathsf{Fake}_2)$ for the relation R_{DH} (see §2); and (ii) A Sigma protocol $\Sigma' = (\mathcal{P}', \mathcal{V}')$ for the relation R_{1nDH} (see §2). Intuitively, the prover starts by computing the first round a of the Sigma protocol Σ. Hence, it commits to message a using the IDTC with a random 1-non-DH tuple $T \in L_{1nDH}$ as instance (*i.e.*, $A = g^\alpha$, $B = g^\beta$ and $C = g \cdot B^\alpha = g^{\alpha \cdot \beta + 1}$ for random $\alpha, \beta \in \mathbb{Z}_q$)), obtaining a commitment $\mathtt{com}$ and decommitment $\mathtt{dec}$. Next, the prover computes the first round a' of the Sigma protocol Σ' for showing that T is indeed a 1-non-DH tuple, and sends $(\mathtt{com}, a', T)$ to the verifier, which replies with a random challenge $c \in \{0,1\}^\ell$. Finally, the prover completes the transcripts of both Σ and Σ' using c as challenge, obtaining values z, z' that are forwarded to the verifier together with the decommitment information $(\mathtt{dec}, a)$ corresponding to commitment $\mathtt{com}$. In the full version we formally prove that the above construction yields to a delayed-input Sigma protocol for R satisfying both adaptive-input special soundness and adaptive-input SHVZK.

4 Adaptive Security of the Fiat-Shamir Transform

Trapdoor Sigma Protocols. Informally, a trapdoor Sigma protocol is a special Sigma protocol in the CRS model satisfying the following two properties: (i) If

[13] Strictly speaking, [43] only achieves a weaker flavor of adaptive-input special soundness that allows to extract the witness for only one of the two theorems.

the statement x is false, then for every first message a, there is a unique challenge c for which there is an accepting third message z that results in an accepting transcript (a, c, z); (ii) There is a trapdoor associated with the CRS that allows us to efficiently compute this "bad challenge" c from the first message a and the statement x being proven. We now slightly revisit the definition of trapdoor Sigma protocols from [14], and show that the Fiat-Shamir transform applied to a trapdoor Sigma protocol (where the hash function takes as input both the statement and the first round of the prover) yields a NIZK with *adaptive soundness*. The only difference between the definition in [14] and ours is that we require the honestly generated CRS to be identically distributed to the CRS generated together with the trapdoor.[14] We also show that, assuming the trapdoor Sigma protocol admits an adaptive-input SHVZK simulator, then the NIZK resulting from the FS transform (where the hash function now takes as input only the first round of the prover) satisfies *adaptive zero knowledge*.

Definition 13 (CRSigma protocols). *We say that a three-round public-coin SHVZK proof system $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$[15] in the CRS model is a CRSigma protocol if for every valid CRS ω, every instance $x \notin L$, and every first round a, there is at most one challenge $c := f(\omega, x, a)$ such that (ω, x, a, c, z) is an accepting transcript for some z. We informally call f the "bad-challenge function" associated to Σ, and note that f may not be efficiently computable.*[16]

Definition 14 (Trapdoor Sigma protocol). *We say that a CRSigma protocol $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ with bad-challenge function f is a* trapdoor Sigma protocol *if there are PPT algorithms* $\mathsf{TrapGen}$, $\mathsf{BadChallenge}$ *with the following syntax:*

- $\mathsf{TrapGen}$ *takes as input the unary representation of the security parameter and outputs a common reference string ω with a trapdoor τ.*
- $\mathsf{BadChallenge}$ *takes as input a trapdoor τ, common reference string ω, an instance x and the first message a and outputs a challenge c.*

We additionally require the following properties.

- ***CRS indistinguishability.*** *An honestly generated common reference string ω is identically distributed to a common reference string output by* $\mathsf{TrapGen}(1^\lambda)$.
- ***Correctness.*** *For every instance $x \notin L$ and for all $(\omega, \tau) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^\lambda)$ we have that $\mathsf{BadChallenge}(\tau, \omega, x, a) = f(\omega, x, a)$.*

The Fiat-Shamir transform. Let $\mathcal{H}$ be a hash family and $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ be a (delayed-input) CRSigma protocol for some relation R. Consider the following non-interactive argument systems $\Pi' = (\mathsf{Gen}', \mathcal{P}', \mathcal{V}')$ and $\Pi'' = (\mathsf{Gen}', \mathcal{P}'', \mathcal{V}')$ for R:

- The common reference string $\omega' := (\omega, k)$ consists of the common reference string of Σ (*i.e.*, $\omega \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda)$) along with a hash key $k \leftarrow\!\!\$\ \mathcal{H}.\mathsf{Gen}(1^\lambda)$.

[14] This modification is related to the fact that we want to prove adaptive soundness (more on this later).

[15] In this case the SHVZK simulator computes also the CRS.

[16] We observe that this notion implies that a trapdoor Sigma protocol is sound with soundness error $2^{-|c|}$.

- Upon input $(x, w) \in R$, the prover $\mathcal{P}'$ (resp. $\mathcal{P}''$) computes $(a, st) \leftarrow\!\!\$\ \mathcal{P}(1^\lambda, \omega, x, w)$ (resp. $(a, st) \leftarrow\!\!\$\ \mathcal{P}(1^\lambda, \omega)$), $c := h_k(a||x)$ (resp. $c := h_k(a)$) and $z \leftarrow\!\!\$\ \mathcal{P}(st, c)$ (resp. $z \leftarrow\!\!\$\ \mathcal{P}(st, x, w, c)$), and outputs[17] (a, c, z).
- The verifier $\mathcal{V}'$ (resp. $\mathcal{V}''$) accepts the transcript (a, c, z) w.r.t. CRS $\omega' = (\omega, k)$ and statement x if $\mathcal{V}(\omega, x, a, c, z) = 1$ and $h_k(a||x) = c$ (resp. $h_k(a) = c$).

Theorem 1. *Suppose that $\mathcal{H}$ is a hash family that is correlationintractable for all sub-exponentially sparse relations that are searchable in time t, and that $\mathcal{H}$ enjoys programmability. Moreover, assume that $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V}, \mathsf{TrapGen}, \mathsf{BadChallenge})$ is a trapdoor Sigma protocol with SHVZK and challenge (second message) space $\{0,1\}^{\lambda^\epsilon}$ for some $\epsilon > 0$, such that $\mathsf{BadChallenge}(\tau, \omega, x, a)$ is computable in time t. Then, the non-interactive argument system Π' described above satisfies zero-knowledge and* adaptive soundness *in the CRS model.*

Theorem 2. *Suppose that $\mathcal{H}$ is a hash family that is correlationintractable for all sub-exponentially sparse relations that are searchable in time t, and that $\mathcal{H}$ enjoys programmability. Moreover, assume that $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V}, \mathsf{TrapGen}, \mathsf{BadChallenge})$ is a trapdoor Sigma protocol with* adaptive-input *SHVZK*[18] *and challenge space $\{0,1\}^{\lambda^\epsilon}$ for some $\epsilon > 0$, such that $\mathsf{BadChallenge}(\tau, \omega, x, a)$ is computable in time t. Then, the non-interactive argument system Π'' described above satisfies soundness and* adaptive zero knowledge *in the CRS model.*

We refer to the full version for the proofs of the theorems 1 and 2

From CRSigma Protocols to Trapdoor Sigma Protocols. In [14], the authors show that a modified version of the protocol for Hamiltonian graphs [31, 43] is a trapdoor Sigma protocol. This allows to obtain a trapdoor Sigma protocol for any NP relation R by just making an NP reduction. In this section we show that *any* CRSigma protocol can be turned into a trapdoor Sigma protocol without making use of expensive NP reductions. Let $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ be a CRSigma protocol for a polynomial-time relation R. Without loss of generality, we assume that the challenge space of Σ is $\{0,1\}$. We construct a trapdoor Sigma protocol $\Sigma' := (\mathsf{Gen}', \mathcal{P}', \mathcal{V}')$ for R based on Σ and on a public-key encryption (PKE) scheme $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ with perfect correctness. The PKE scheme is essentially used as a commitment, similarly to what is done in [14]. At a high level our transform works as follows. The CRS consists of a public key for the PKE scheme and of a CRS for Σ. To compute a proof, the prover generates the first message of Σ and the replies to the challenge 0 and 1 that we denote respectively with z_0 and z_1. Then, the prover encrypts z_0 and z_1 and sends these encrypted values together with the first round of Σ to the verifier. The verifier sends a random bit c, and the prover replies with z_c and the randomness

[17] Equivalently, the prover can just output (a, z) as c can be re-computed by the verifier.
[18] As in the definition of adaptive-input SHVZK, the simulator is defined by two algorithms $(\mathcal{S}_0, \mathcal{S}_1)$. The difference is that $\mathcal{S}_0$ outputs the CRS in addition.

used to compute the encryption of z_c. Finally, the verifier accepts if the randomness and the value z_c are consistent with the commitment received in the first round and if the transcript for Σ is accepting. We note that given the secret key of the encryption scheme it is possible to extract the bad challenge (if any). And this is the intuitive reason why our protocol is a trapdoor Sigma protocol. We refer to the full version for the formal description of the protocol and the proof. We remark that it is always possible to extend the challenge space of the above protocol to $\{0,1\}^\kappa$ for any $\kappa \in \mathbb{N}$ without compromising its completeness, by just repeating it in parallel κ times. Then, using Theorem 1, we obtain an *adaptively-sound* NIZK.

Adding Adaptive-Input SHVZK. To transform a delayed-input trapdoor Sigma protocol $\Sigma = (\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ into one with adaptive-input SHVZK Σ'' we follow the same approach proposed in §3. The prover computes the first round of Σ and commits to it using an IDTC that enjoys a special form of extractability. We refer the reader to the full version for the formal definition of extractable IDTCs, its concrete instantiation based on the DDH assumption, and for the formal description of Σ'' with its security analysis.

References

1. Ananth, P., Choudhuri, A.R., Jain, A.: A new approach to round-optimal secure multiparty computation. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 468–499. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63688-7˙16
2. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4˙29
3. Badrinarayanan, S., Goyal, V., Jain, A., Kalai, Y.T., Khurana, D., Sahai, A.: Promise zero knowledge and its applications to round optimal MPC. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 459–487. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0˙16
4. Barak, B.: How to go beyond the black-box simulation barrier. In: 42nd FOCS. pp. 106–115. IEEE Computer Society Press (Oct 2001). https://doi.org/10.1109/SFCS.2001.959885
5. Barak, B., Lindell, Y., Vadhan, S.P.: Lower bounds for non-black-box zero knowledge. In: 44th FOCS. pp. 384–393. IEEE Computer Society Press (Oct 2003). https://doi.org/10.1109/SFCS.2003.1238212
6. Bellare, M., Goldreich, O.: On defining proofs of knowledge. In: Brickell, E.F. (ed.) CRYPTO'92. LNCS, vol. 740, pp. 390–420. Springer, Heidelberg (Aug 1993). https://doi.org/10.1007/3-540-48071-4˙28
7. Bellare, M., Ristov, T.: A characterization of chameleon hash functions and new, efficient designs. Journal of Cryptology **27**(4), 799–823 (Oct 2014). https://doi.org/10.1007/s00145-013-9155-8

8. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 93. pp. 62–73. ACM Press (Nov 1993). https://doi.org/10.1145/168588.168596
9. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4˙38
10. Bitansky, N., Dachman-Soled, D., Garg, S., Jain, A., Kalai, Y.T., López-Alt, A., Wichs, D.: Why "Fiat-Shamir for proofs" lacks a proof. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 182–201. Springer, Heidelberg (Mar 2013). https://doi.org/10.1007/978-3-642-36594-2˙11
11. Blum, M.: How to prove a theorem so no one else can claim it. In: In Proceedings of the International Congress of Mathematicians. p. 444–451 (1986)
12. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four round secure computation without setup. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 645–677. Springer, Heidelberg (Nov 2017). https://doi.org/10.1007/978-3-319-70500-2˙22
13. Brakerski, Z., Koppula, V., Mour, T.: NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. IACR Cryptology ePrint Archive **2020**, 258 (2020), https://eprint.iacr.org/2020/258
14. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-Shamir: from practice to theory. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC. pp. 1082–1090. ACM Press (Jun 2019). https://doi.org/10.1145/3313276.3316380
15. Canetti, R., Chen, Y., Reyzin, L.: On the correlation intractability of obfuscated pseudorandom functions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 389–415. Springer, Heidelberg (Jan 2016). https://doi.org/10.1007/978-3-662-49096-9˙17
16. Canetti, R., Chen, Y., Reyzin, L., Rothblum, R.D.: Fiat-Shamir and correlation intractability from strong KDM-secure encryption. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 91–122. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9˙4
17. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: 30th ACM STOC. pp. 209–218. ACM Press (May 1998). https://doi.org/10.1145/276698.276741
18. Catalano, D., Visconti, I.: Hybrid trapdoor commitments and their applications. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 298–310. Springer, Heidelberg (Jul 2005). https://doi.org/10.1007/11523468˙25
19. Choudhuri, A.R., Ciampi, M., Goyal, V., Jain, A., Ostrovsky, R.: Round optimal secure multiparty computation from minimal assumptions. Cryptology ePrint Archive, Report 2019/216 (2019), https://eprint.iacr.org/2019/216
20. Ciampi, M., Ostrovsky, R., Siniscalchi, L., Visconti, I.: Concurrent non-malleable commitments (and more) in 3 rounds. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 270–299. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53015-3˙10
21. Ciampi, M., Ostrovsky, R., Siniscalchi, L., Visconti, I.: Four-round concurrent non-malleable commitments from one-way functions. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 127–157. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63715-0˙5

22. Ciampi, M., Ostrovsky, R., Siniscalchi, L., Visconti, I.: Round-optimal secure two-party computation from trapdoor permutations. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 678–710. Springer, Heidelberg (Nov 2017). https://doi.org/10.1007/978-3-319-70500-2˙23
23. Ciampi, M., Persiano, G., Scafuro, A., Siniscalchi, L., Visconti, I.: Online/offline OR composition of sigma protocols. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 63–92. Springer, Heidelberg (May 2016). https://doi.org/10.1007/978-3-662-49896-5˙3
24. Ciampi, M., Persiano, G., Siniscalchi, L., Visconti, I.: A transform for NIZK almost as efficient and general as the Fiat-Shamir transform without programmable random oracles. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part II. LNCS, vol. 9563, pp. 83–111. Springer, Heidelberg (Jan 2016). https://doi.org/10.1007/978-3-662-49099-0˙4
25. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y. (ed.) CRYPTO'94. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48658-5˙19
26. Damgård, I.: On Σ-protocol. http://www.cs.au.dk/~ivan/Sigma.pdf (2010)
27. Damgård, I., Groth, J.: Non-interactive and reusable non-malleable commitment schemes. In: 35th ACM STOC. pp. 426–437. ACM Press (Jun 2003). https://doi.org/10.1145/780542.780605
28. Damgård, I., Nielsen, J.B.: Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 581–596. Springer, Heidelberg (Aug 2002). https://doi.org/10.1007/3-540-45708-9˙37
29. Dwork, C., Naor, M., Reingold, O., Stockmeyer, L.J.: Magic functions. In: 40th FOCS. pp. 523–534. IEEE Computer Society Press (Oct 1999). https://doi.org/10.1109/SFFCS.1999.814626
30. Faust, S., Kohlweiss, M., Marson, G.A., Venturi, D.: On the non-malleability of the Fiat-Shamir transform. In: Galbraith, S.D., Nandi, M. (eds.) INDOCRYPT 2012. LNCS, vol. 7668, pp. 60–79. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34931-7˙5
31. Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In: 31st FOCS. pp. 308–317. IEEE Computer Society Press (Oct 1990). https://doi.org/10.1109/FSCS.1990.89549
32. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7˙12
33. Fischlin, M., Fischlin, R.: The representation problem based on factoring. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 96–113. Springer, Heidelberg (Feb 2002). https://doi.org/10.1007/3-540-45760-7˙8
34. Goldwasser, S., Kalai, Y.T.: On the (in)security of the Fiat-Shamir paradigm. In: 44th FOCS. pp. 102–115. IEEE Computer Society Press (Oct 2003). https://doi.org/10.1109/SFCS.2003.1238185
35. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: 17th ACM STOC. pp. 291–304. ACM Press (May 1985). https://doi.org/10.1145/22145.22178
36. Goyal, V., Richelson, S.: Non-malleable commitments using Goldreich-Levin list decoding. In: Zuckerman, D. (ed.) 60th FOCS. pp. 686–699. IEEE Computer Society Press (Nov 2019). https://doi.org/10.1109/FOCS.2019.00047

37. Goyal, V., Richelson, S., Rosen, A., Vald, M.: An algebraic approach to non-malleability. In: 55th FOCS. pp. 41–50. IEEE Computer Society Press (Oct 2014). https://doi.org/10.1109/FOCS.2014.13
38. Halevi, S., Myers, S., Rackoff, C.: On seed-incompressible functions. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 19–36. Springer, Heidelberg (Mar 2008). https://doi.org/10.1007/978-3-540-78524-8˙2
39. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols - Techniques and Constructions. Information Security and Cryptography, Springer (2010)
40. Holmgren, J., Lombardi, A.: Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In: Thorup, M. (ed.) 59th FOCS. pp. 850–858. IEEE Computer Society Press (Oct 2018). https://doi.org/10.1109/FOCS.2018.00085
41. Kalai, Y.T., Rothblum, G.N., Rothblum, R.D.: From obfuscation to the security of Fiat-Shamir for proofs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 224–251. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63715-0˙8
42. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. pp. 335–354 (2004). https://doi.org/10.1007/978-3-540-28628-8˙21, http://dx.doi.org/10.1007/978-3-540-28628-8_21
43. Lapidot, D., Shamir, A.: Publicly verifiable non-interactive zero-knowledge proofs. In: Menezes, A.J., Vanstone, S.A. (eds.) CRYPTO'90. LNCS, vol. 537, pp. 353–365. Springer, Heidelberg (Aug 1991). https://doi.org/10.1007/3-540-38424-3˙26
44. Lindell, Y.: An efficient transform from sigma protocols to NIZK with a CRS and non-programmable random oracle. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 93–109. Springer, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46494-6˙5
45. Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (Dec 2009). https://doi.org/10.1007/978-3-642-10366-7˙35
46. Mittelbach, A., Venturi, D.: Fiat-Shamir for highly sound protocols is instantiable. In: Zikas, V., De Prisco, R. (eds.) SCN 16. LNCS, vol. 9841, pp. 198–215. Springer, Heidelberg (Aug / Sep 2016). https://doi.org/10.1007/978-3-319-44618-9˙11
47. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO'92. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (Aug 1993). https://doi.org/10.1007/3-540-48071-4˙3
48. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7˙4
49. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (May 1996). https://doi.org/10.1007/3-540-68339-9˙33
50. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34805-0˙22
51. Stern, J.: A new identification scheme based on syndrome decoding. In: Stinson, D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48329-2˙2